

**计算机与信息技术学院**

**课程设计报告**

|  |  |  |  |
| --- | --- | --- | --- |
|  | 课程名称 | 软件系统设计 |  |
|  | 设计名称 | 基于AI的省钱助手 |  |
|  | 专业班级 | 计科2101 |  |
|  | 学 号 | 21851300 |  |
|  | 姓 名 | 杨晨 |  |
|  | 任课教师 | 田晋坤 |  |
|  | 日 期 | 2024-06-27 |  |

**《软件系统设计》课程考核说明**

**一、考试对象**

2021级计科2101-2106班

**二、考核方式**

课程设计报告 + 答辩。

**三、考试内容**

基于大语言模型（LLM）完成一个智能化系统，使用的技术包括但不限于Python、Langchain等。其中，大语言模型可以采用现有的第三方服务商提供的模型（如：通义千问、千帆等），也可以使用开源模型。

**四、技术要求**

1、要求使用Python、Langchain框架。

2、项目文件命名：学生姓名的拼音首字母-网站英文名称，名称与多个英文单词之间用“-”连接，如李明的手机销售网站命名为：lm-phone-seller

3、单文件组件命名要符合Pascal命名规范，变量、文件及文件夹命名要符合规范，尽量做到见名知意。

**五、提交物**

项目源码、设计说明书电子版、设计说明书纸质打印版、答辩纪录。

**六、期末成绩构成**

由项目成绩和答辩成绩组成，其中项目成绩60分，答辩成绩40分

项目成绩评分标准如下：

|  |  |
| --- | --- |
| **综合/单项类评分** | **分值** |
| **1 提交物** | **10** |
| 1.1 内容完整性 | 5 |
| 1.2 目录结构 | 5 |
| **2 代码** | **20** |
| 2.1 项目、代码命名规范 | 5 |
| 2.2 代码编写规范性 | 5 |
| 2.3 代码可读性 | 5 |
| 2.4 技术全面 | 5 |
| **3 程序演示** | **30** |
| 3.1 结果正确，功能完整 | 15 |
| 3.2 页面设计美观 | 15 |

**《软件系统设计》课程考核成绩**

|  |  |
| --- | --- |
| 项目成绩（60分） |  |
| 答辩成绩（40分） |  |
| 期末总成绩 |  |
| **答辩评语：**  提交物内容（完整、较完整、基本完整、不完整），目录结构（正确、较正确、基本正确、不正确）；项目、代码命名（符合规范、较符合规范、基本符合规范、不规范），代码注释（完整、较完整、基本完整、不完整），可读性（好、较好、尚可、差）；技术应用方面（全面、较全面、基本合格、有欠缺）；程序功能（完整、较完整、基本完整、不完整），页面（美观、较美观、设计尚可、不美观），演示过程（顺畅、较顺畅、基本顺畅、不顺畅），问题回答（正确、较正确、基本正确、错误较多），思路（清晰、较清晰、基本清晰、不清晰），概念（清楚、较清楚、基本清楚、不清楚）。 | |

**目 录**

[第1章 产品需求 1](#_Toc4959)

[1.1 背景和目的 1](#_Toc20144)

[1.2 用户需求和系统功能 2](#_Toc27172)

[1.3 产品设计和页面 3](#_Toc4991)

[1.4 项目资源和进度安排 3](#_Toc14456)

[第2章 概要设计 4](#_Toc27103)

[2.1 架构设计 4](#_Toc10852)

[2.2 功能设计 5](#_Toc2454)

[2.3 模型设计 6](#_Toc27889)

[第3章 系统测试 11](#_Toc22135)

[3.1 测试范围 11](#_Toc23504)

[3.2 测试用例 12](#_Toc20472)

[第4章 总结 14](#_Toc18712)

[《软件系统设计》课程答辩记录 15](#_Toc30305)

第1章 产品需求

## 1.1 背景和目的

### 1.1.1 产品背景

随着人工智能技术的快速发展，AI在各个领域的应用越来越广泛。在消费领域，AI可以帮助消费者做出更加明智的决策。然而，市场上现有的AI产品多集中在个性化推荐和促销活动上，较少关注消费者的长期财务健康和消费行为的可持续性。因此，开发一款基于AI问答的省钱助手，不仅符合技术发展趋势，也满足了市场对于理性消费工具的需求。

本系统的实现旨在帮助用户在购买商品前给用户给出商品购买前的多方面考量，避免用户做出冲动消费行为。

### 1.1.2 产品目标

省钱助手问答机器人通过结合人工智能技术和用户友好的界面，旨在帮助用户在消费决策时更加理智和审慎，从而实现个人和社会层面的多重益处。

**1) 提升消费意识**

* 利用人工智能技术，提供基于用户输入的个性化消费建议，帮助用户识别和抑制不必要的购买欲望。
* 目标：用户在每次使用服务后，至少能识别出一项原本可能冲动购买的商品。

**2) 增强个人财务管理**

* 通过提供消费理由分析，帮助用户更好地理解自己的消费行为，从而做出更明智的财务决策。
* 目标：用户在使用服务一年后，平均减少非必要支出达到15%。

**3) 提高用户满意度和忠诚度**

* 通过自然语言处理技术，优化输出的表达方式，确保用户获得清晰、友好的交互体验。
* 目标：在服务上线后的第一年内，用户满意度提升至90%以上。

**4) 拓展服务范围**

* 将省钱助手问答机器人服务推广至不同的平台和应用场景，如移动应用、社交媒体等，以覆盖更广泛的用户群体。
* 目标：在服务上线后的两年内，用户基础增长至少50%。

通过实现上述目标，省钱助手问答机器人项目旨在为用户提供一个高效、智能的消费决策辅助工具，帮助他们在消费过程中做出更加理智的选择，同时为社会带来积极的经济和环境效益。

## 1.2 用户需求和系统功能

用户作为主要参与者，与系统进行交互。首先输入他们考虑购买的产品信息。系统接收这些信息，并使用内置的人工智能模型来生成抑制购买的理由。用户可以查看这些理由，并评估它们以决定是否继续购买。系统还提供自然语言处理服务，以改善输出的表达方式，使其更加自然和易于理解。省钱助手系统用例说明表如图1-1所示。

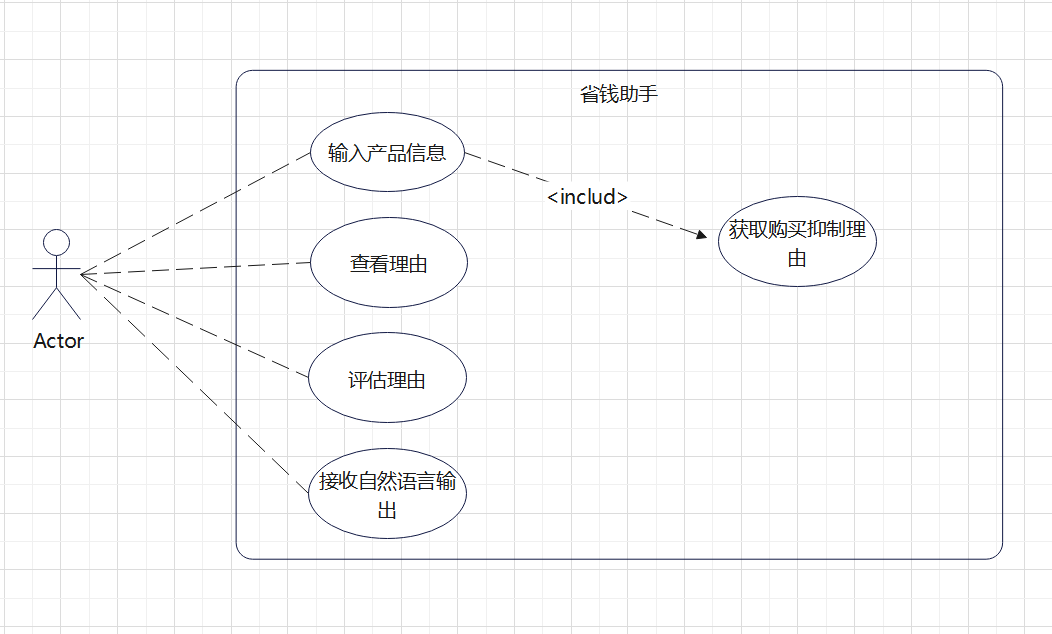


图1-1 系统用例图

## 1.3 产品设计和页面

### 1.3.1 问答窗口

用户进入省钱助手页面在问答窗口键入物品，系统便会在下方给出抑制购买的理由。页面展示如图1-2所示。

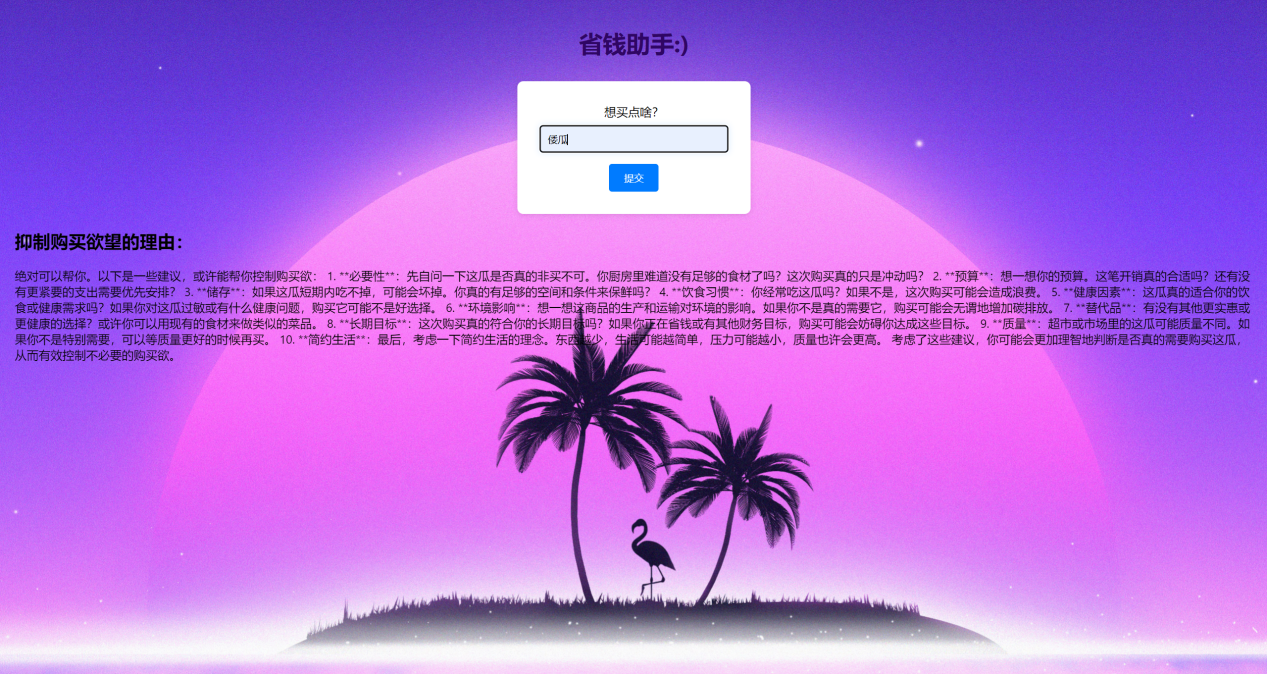


图1-2 用户管理界面展示图

## 1.4 项目资源和进度安排

本项目的主要任务包含：调用AI，使用langchain框架组件实现自定义对话。、前端登录跳转，界面设计，前后信息交互。

参与本项目的人员包括：杨晨，卢梦豪。

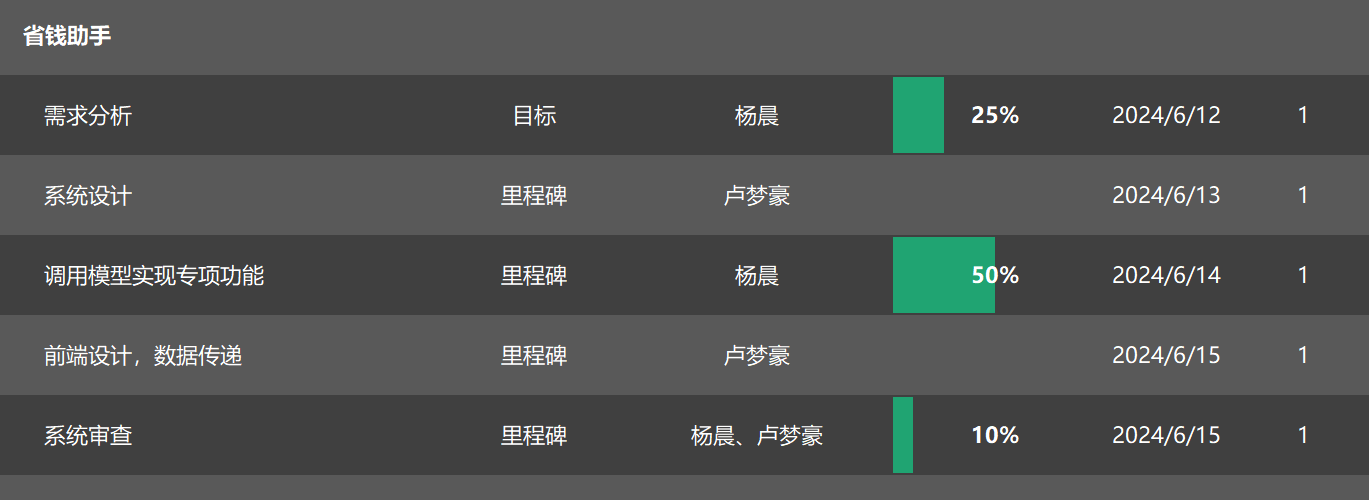


图1-3 进度甘特图

# 第2章 概要设计

## 2.1 架构设计

### 2.1.1 功能架构

省钱助手问答机器人项目的功能架构是一个集成了用户界面、后端逻辑和人工智能处理的系统。它通过一个简洁的用户界面接收用户输入的产品信息，然后通过后端将这些信息传递给AI模型，该模型生成并优化购买抑制理由。优化后的理由被反馈到用户界面供用户参考，以帮助他们重新考虑购买决策。整个过程旨在提供流畅、直观且互动性强的用户体验。而通过功能架构图，清晰的展现出系统要完成的任务。功能架构图具体设计如图2-1所示。

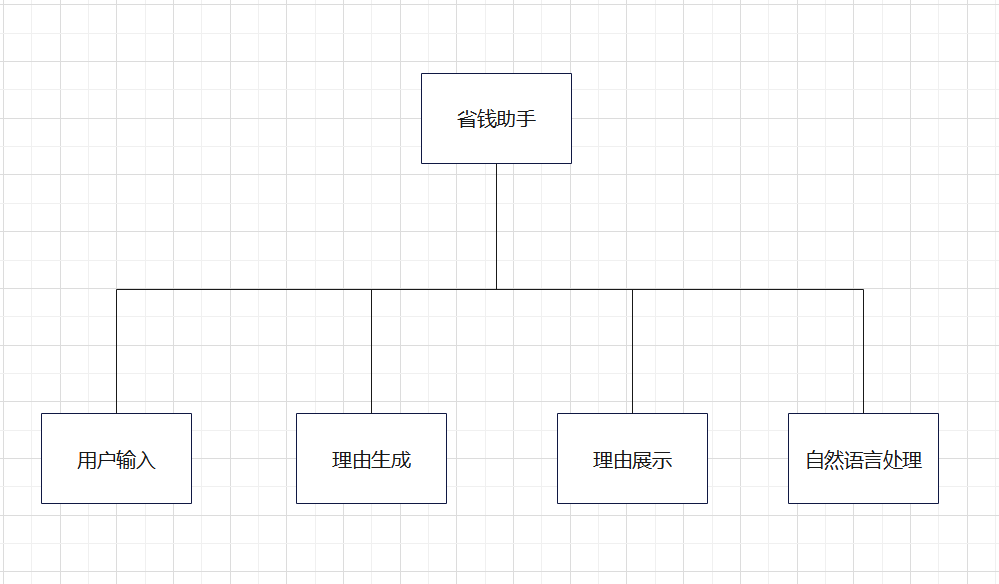


图2-1 功能架构设计图

### 2.1.2 技术架构

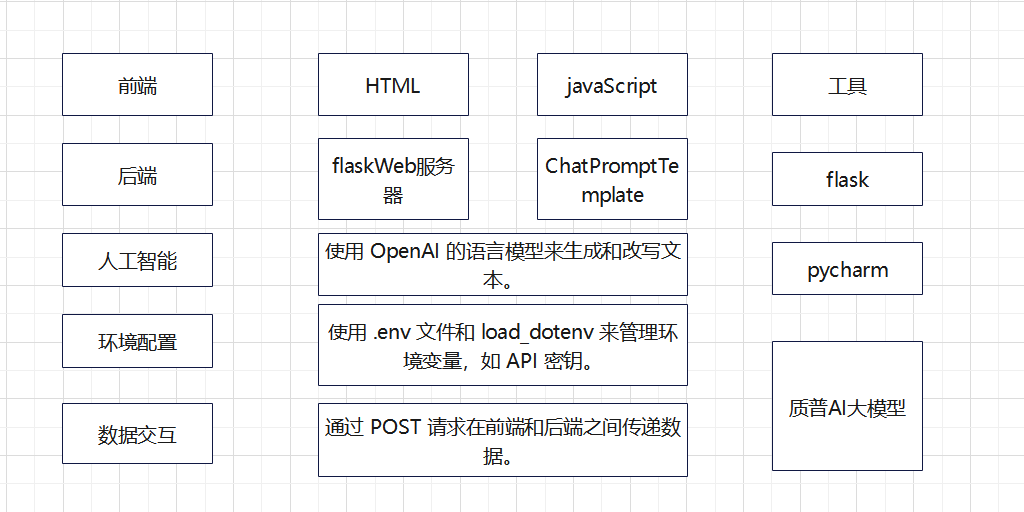


图2-2 技术架构设计图

## 2.2 功能设计

后端 (Python Flask 应用程序)

* 初始化 Flask 应用：创建 Flask 实例，加载环境变量。
* 定义 ChatPromptTemplate：设置与用户交互的提示模板。
* 配置 ChatOpenAI 模型：设置用于生成回答的 AI 模型参数。
* 输出解析函数：定义一个函数，用于改写 AI 生成的文本，使其更自然。
* 创建路由和视图：设置路由，根据请求方法处理 GET 或 POST 请求。如果是 POST 请求，从表单获取产品名称，并调用 AI 模型生成回答。
* 运行 Flask 应用：启动 Flask 服务器。

### 2.2.1 问答交互

省钱助手问答机器人的使用流程是直观而连贯的：用户访问应用程序的界面，输入他们正在考虑购买的产品名称，随后提交表单。系统接收到用户输入后，通过集成的AI模型处理并生成一系列抑制购买欲望的理由。这些理由经过自然语言处理优化，以提高文本的可读性和说服力，然后反馈给用户，帮助他们重新考虑其购买决策。整个过程旨在提供快速、高效的用户支持，使用户能够做出更加理智的消费选择。

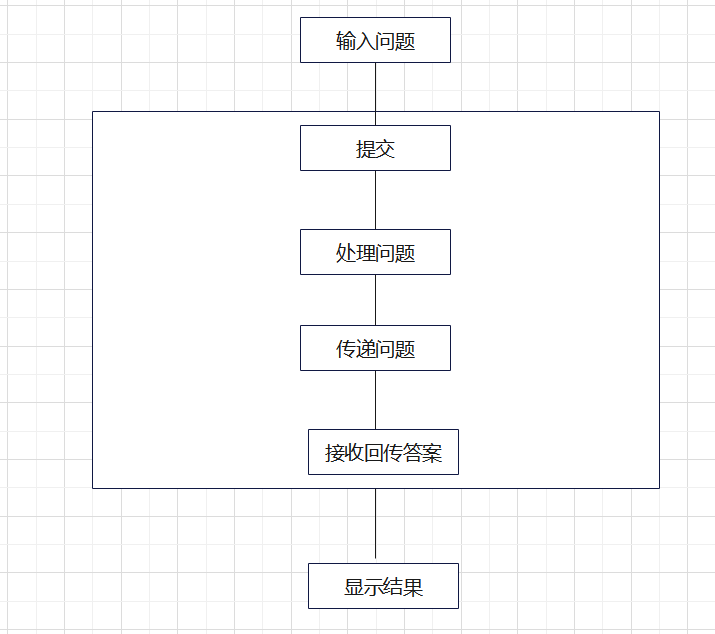


图2-3 问答流程图

## 2.3 模型设计

### 2.3.1 概念模型

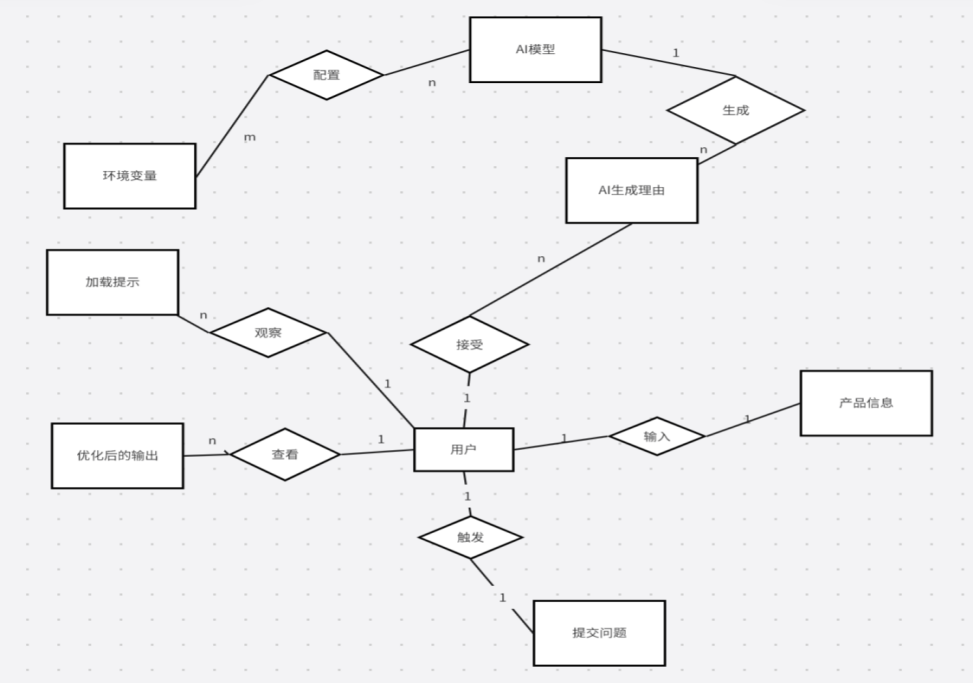


图2-4 E-R图

**关系和多重性：**

* 用户与产品信息：

关系：输入产品信息

多重性：一个用户可以输入多个产品信息，但每个产品信息只能由一个用户输入。

* 用户与提交查询：

关系：提交

多重性：一个用户可以提交多个查询。

* 用户与AI生成理由：

关系：接收理由

多重性：一个用户可以接收多个AI生成理由。

* 用户与优化后的输出：

关系：查看输出

多重性：一个用户可以查看多个优化后的输出。

* 提交查询与AI生成理由：

关系：生成

多重性：一个提交可以生成多个AI理由。

* AI生成理由与优化后的输出：

关系：优化

多重性：一个AI理由可以被优化为一个优化后的输出。

* 环境变量与AI模型：

关系：配置

多重性：多个环境变量可以配置一个AI模型。

* AI模型与AI生成理由：

关系：生成理由

多重性：一个AI模型可以生成多个AI理由。

### 2.3.2 逻辑模型

为省钱助手项目设计逻辑模型，需要考虑数据结构、系统组件以及它们之间的交互。以下是设计逻辑模型如下图2-5：

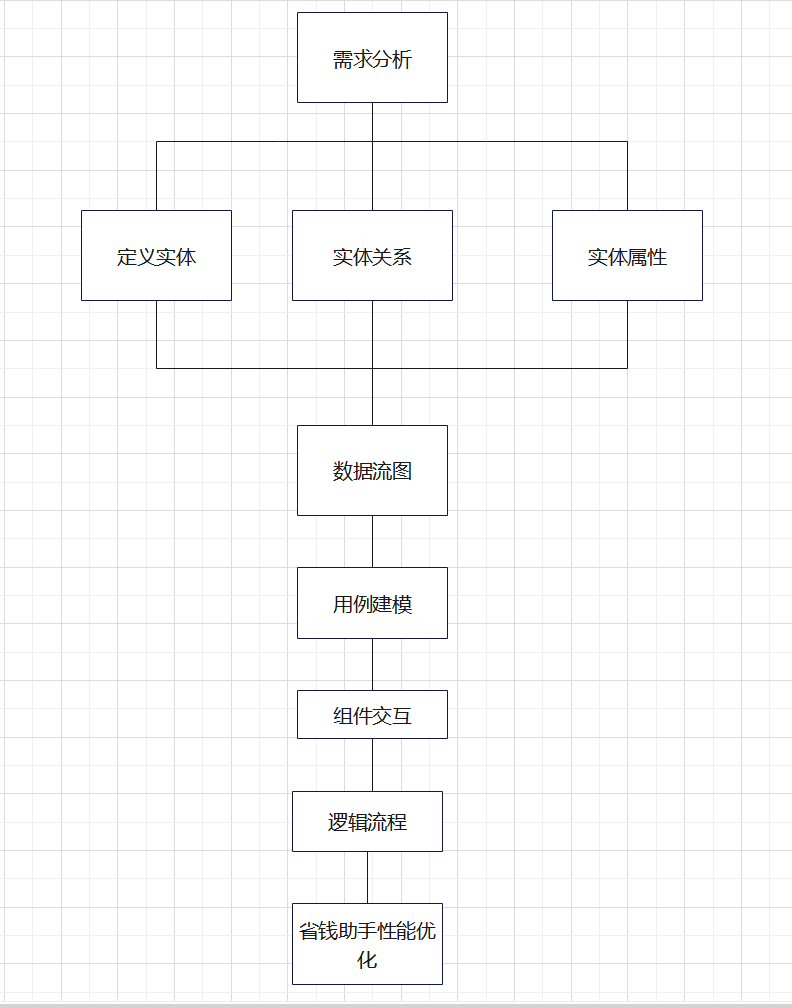


图2-5 逻辑模型图

### 2.3.3 物理模型

### Web服务器：一个或多个Web服务器用于托管Flask应用程序，处理HTTP请求和响应。

### 应用服务器： Flask应用程序运行在应用服务器上，该服务器负责处理业务逻辑和与AI模型的交互。

### 人工智能服务： 一个或多个AI服务实例，提供语言处理能力，可能是远程API调用或本地部署的服务。

### 网络结构：用户通过互联网访问Web服务器，服务器之间可能通过内部网络进行通信。

### 前端资源： 前端HTML、CSS和JavaScript文件可能存储在静态资源服务器或内容分发网络（CDN）上。

### 外部API服务：如果您的应用需要调用外部API（如OpenAI API），这些服务将作为外部依赖存在。

### 监控和日志系统： 用于监控服务器状态和记录应用日志的系统，帮助维护和故障排查。

### 用户访问：用户通过浏览器访问前端资源，与应用服务器进行交互。

### 部署环境： 应用可能部署在云环境或本地。

### 这个物理模型是基于代码的功能和常见的Web应用部署实践结果。

### 2.3.1 整体接口设计

由省钱助手项目的前端和后端代码，我们设计了一套详细的接口规范。接口设计通常包括请求的路径、支持的操作、请求和响应的数据格式等。以下是省钱助手项目可能的接口设计：

1. 首页接口

路径：/

方法：GET

描述：展示省钱助手的首页，包括用户输入界面。

请求参数：无

响应：成功：HTML页面，包含表单和加载提示。

失败：错误页面。

2. 提交产品信息接口

路径：/

方法：POST

描述：接收用户输入的产品名称，并请求AI生成抑制购买欲望的理由。

请求参数：product：用户输入的产品名称，类型为string。

响应：成功：HTML页面，包含生成的理由。

失败：错误信息。

3. AI理由生成接口

路径：/generate-reasons

方法：POST

描述：后端接口，接收产品名称并调用AI模型生成理由。

请求体：{

"product": "string"

}

响应：{

"success": true,

"content": "string" // AI生成的理由文本

}

或{

"success": false,

"error": "string" // 错误信息

}

4. 自然语言处理接口

路径：/nlp-process

方法：POST

描述：接收AI生成的原始理由文本，返回更自然、流畅的文本。

请求体：{

"text": "string"

}

响应：{

"success": true,

"content": "string" // 自然语言处理后的文本

}

或复制{

"success": false,

"error": "string" // 错误信息

}

5. 清理输入接口

路径：/clear-input

方法：POST

描述：清空用户输入的产品名称，通常在表单提交后调用。

请求参数：无

响应：成功：无内容或重定向。

失败：错误信息。

# 第3章 系统测试

## 3.1 测试范围

1. 功能测试

* 用户输入验证：测试输入字段是否接受有效和无效的输入。验证必填项是否在表单提交时进行检查。
* 表单提交：测试表单在正常和异常情况下的提交行为。验证提交后数据是否正确发送到服务器。
* AI理由生成：测试AI模型是否根据用户输入生成合理的购买抑制理由。检查生成理由的准确性和相关性。
* 自然语言处理：验证输出文本是否经过自然语言处理，提高可读性。
* 结果展示：测试结果页面是否正确显示生成的理由。验证在不同条件下（如无理由生成时）的用户界面表现。
* 用户界面交互：测试按钮、输入框等元素的交互是否符合预期。

2. 非功能测试

* 性能测试：测试应用在高负载下的表现，包括响应时间和服务器处理能力。
* 安全性测试：验证数据传输安全性，检查是否有适当的加密措施。
* 可用性测试：确保应用的易用性，包括导航、布局和用户指导。
* 兼容性测试：测试应用在不同浏览器和设备上的表现。
* 可访问性测试：确保应用符合可访问性标准，使所有用户都能使用。

3. 集成测试

* 测试前端和后端的集成是否无缝。验证外部API（如OpenAI服务）与应用的集成情况。

4. 系统测试

* 测试整个应用作为一个系统运行时的表现。验证所有组件作为一个整体协同工作的结果。

5. 回归测试

* 在每次更新后执行，确保新代码没有破坏现有功能。

6. 用户接受测试（UAT）

* 让实际用户测试应用，收集反馈以验证是否满足用户需求。

## 3.2 测试用例

基于上述测试范围，以下是省钱助手项目的测试用例示例：

1. 用户输入验证测试用例

用例ID：TIU\_01

目的：验证输入字段接受有效和无效的输入。

前提条件：用户界面已加载，输入字段可见。

测试步骤：

1. 清空输入字段。

2. 输入一个常见的产品名称（有效输入）。

3. 提交表单。

4. 验证系统是否接受输入并响应。

5. 清空输入字段。

6. 输入特殊字符或空格（无效输入）。

7. 提交表单。

8. 验证系统是否显示错误消息或拒绝提交。

预期结果：系统应接受有效输入并生成响应，拒绝无效输入并提示错误。

2. 表单提交测试用例

用例ID：TSUB\_01

目的：测试表单提交行为。

前提条件：用户已在输入字段中输入产品名称。

测试步骤：

1. 提交表单。

2. 验证是否显示加载提示。

3. 等待响应完成。

4. 验证是否显示生成的理由。

预期结果：系统应显示加载提示，之后显示生成的理由。

3. AI理由生成测试用例

用例ID：TAI\_01

目的：验证AI模型生成的理由的相关性。

前提条件：用户已提交有效产品名称。

测试步骤：

1. 提交包含产品名称的表单。

2. 验证AI生成的理由是否与产品相关。

预期结果：生成的理由应与输入的产品名称相关。

4. 自然语言处理测试用例

用例ID：TNLP\_01

目的：验证输出文本的自然语言处理效果。

前提条件：AI已生成理由。

测试步骤：

1. 提交表单。

2. 获取AI生成的理由。

3. 验证理由的流畅性和自然性。

预期结果：理由应读起来自然，无语法错误或不流畅的句子。

5. 结果展示测试用例

用例ID：TRES\_01

目的：测试结果的正确展示。

前提条件：AI已生成理由。

测试步骤：

1. 提交包含产品名称的表单。

2. 验证结果页面是否正确显示生成的理由。

预期结果：结果页面应清晰显示生成的理由。

6. 性能测试用例

用例ID：TPERF\_01

目的：测试应用的响应时间。

前提条件：系统在正常运行状态。

测试步骤：

1. 测量提交表单到显示结果的总时间。

2. 在高并发条件下重复测试。

预期结果：系统应在预定时间内响应用户请求，即使在高负载下。

7. 安全性测试用例

用例ID：TSEC\_01

目的：验证数据传输的安全性。

前提条件：用户正在通过HTTPS提交表单。

测试步骤：

1. 提交包含敏感信息的表单。

2. 验证数据是否通过加密传输。

预期结果：所有数据应通过加密通道安全传输。

# 第4章 总结

经过本次《企业级应用开发》课程的学习，在开发省钱助手项目的过程中，我深刻体会到了从概念到实现的全过程。项目始于对用户需求的深入分析，明确了开发一个旨在帮助用户抑制冲动购买的智能助手的目标。用户体验设计成为了项目的核心，我投入大量精力确保界面直观易用，并且做了大量的本地测试。

在技术实现方面，我采用了前后端分离的架构，这不仅提升了开发效率，也便于团队成员之间的协作。API设计成为了连接前端界面和后端逻辑的桥梁，我确保了API的清晰和一致性.测试是项目中不可或缺的一环，我实施了全面的测试策略，包括功能测试、性能测试、安全性测试等，以确保应用的稳定性和可靠性。此外，通过持续集成和持续部署的实践，在错误处理和日志记录方面，我建立了一套有效的机制，以便于问题的快速定位和解决。

技术选型上，我根据项目需求和团队的技术栈进行了审慎的决策。团队协作方面，我采用了敏捷开发方法，通过定期的沟通和任务分配，确保了项目的顺利进展。

最后，我意识到了可维护性和扩展性在系统设计中的重要性，这有助于降低长期维护成本，并为未来可能的功能扩展做好准备。通过省钱助手项目，我获得了宝贵的实践经验，这些经验将指导我在未来项目中的规划和执行。

# 《软件系统设计》课程答辩记录

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 专业班级 | 计科2101 | 姓名 | 杨晨 | 学号 | 21851300 |
| 答辩时间 | 2024-06-17 11：10-12：00 | | 答辩地点 | 思源楼424 | |
| 答辩题目 | 软件系统设计 | | | | |
| 答辩老师 | 田晋坤 | | | | |
| 提问及答辩记录：   1. 如何实现自定义对话？   答：首先加载环境变量，然后定义了一个对话提示模板，该模板指导AI如何根据用户输入生成回应。当用户通过Web表单提交他们考虑购买的产品名称时，Flask后端接收这个输入，并将其传递给定义好的AI模型。AI模型根据预定义的提示和用户输入生成一系列理由，然后通过output\_parser函数进一步优化这些理由的表述。  最后，优化后的理由被发送回前端，并显示给用户，使用户能够看到抑制购买欲望的各种理由。整个过程通过Web界面与用户交互，实现了一个流畅的自定义对话体验。  2.项目采用了哪些技术？  答：1）使用了Python语言和Flask框架来创建Web服务，允许用户通过Web界面与应用交互。  2）利用了dotenv库来管理环境变量，确保API密钥和配置的安全。集成了自定义模块langchain\_openai和langchain.prompts，这些模块封装了与OpenAI API的交互，提供了创建提示模板和调用AI模型的能力。  3）通过ChatPromptTemplate定义了对话的上下文和结构，确保AI生成的回应与用户查询相关。  4）使用了智普AI的语言模型，这个模型能够根据给定的提示生成自然语言回应。  5）实现了一个自定义函数output\_parser，它调用另一个AI模型来优化和改写AI生成的文本，提高其自然性和可读性。 | | | | | |